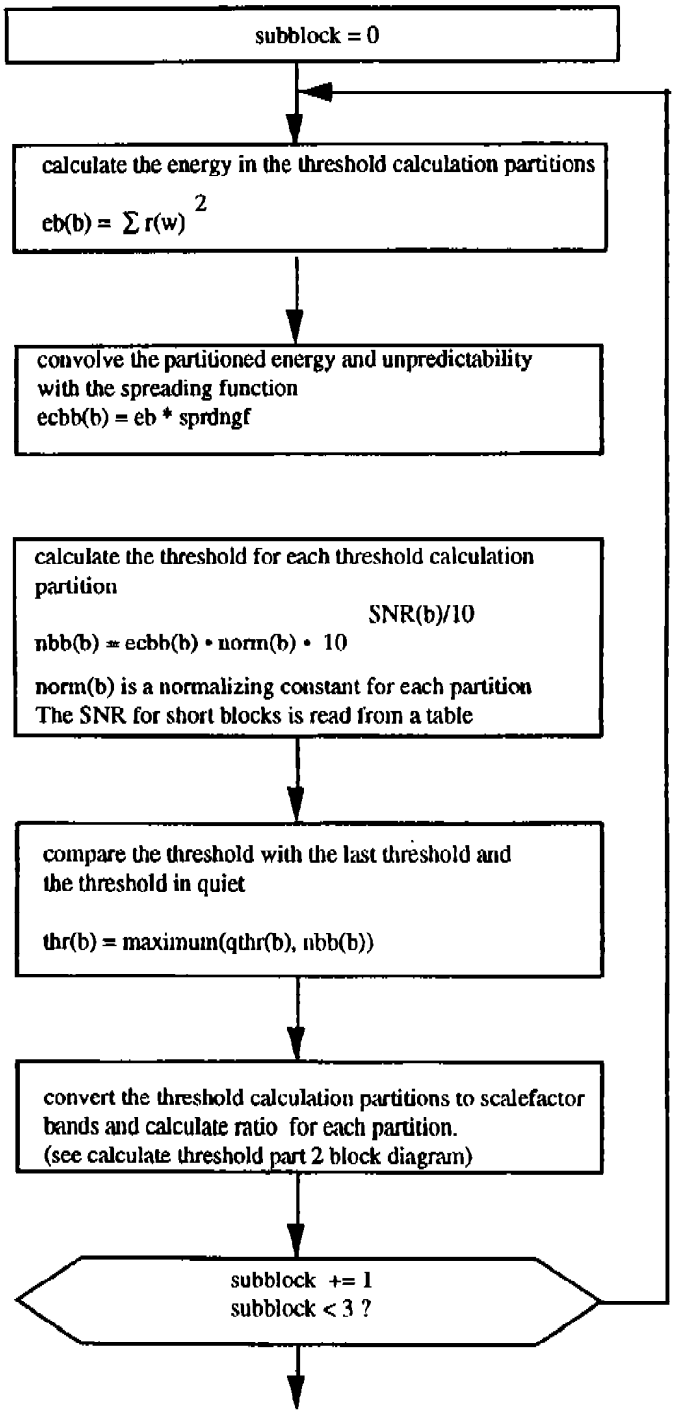

Figure C.6.d -- Block diagram psychoacoustic model 2, Layer III: calculate threshold for short blocks

Window switching decision:

The decision whether the filterbank should be switched to short windows is derived from the calculation of the masking threshold by calculating the estimate of the psychoacoustic entropy (PE) and switching when the PE exceeds the value 1800. If this condition is met, the sequence start (block_type=1), short (block_type=2), short, stop (block_type=3) is started. Figure C.7 shows the possible state changes for the window switching logic.

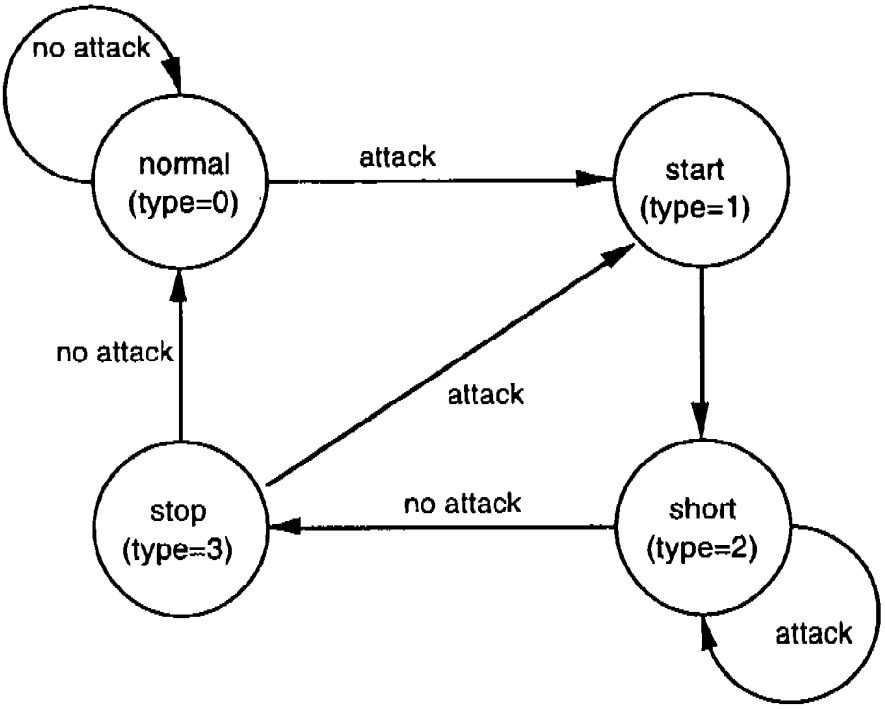

Figure C.7 -- Window Switching State Diagram

C.1.5.3.3 Analysis part of the hybrid filterbank

The subband analysis of the polyphase filterbank is described in clause C.1.3, "Subband analysis filter". The output of the polyphase filterbank is the input to the subdivision using the MDCT. According to the output of the psychoacoustic model (variables **blocksplit_flag** and **block_type**) the window and transform types **normal**, **start**, **short** or **stop** are used.

18 consecutive output values of one granule and 18 output values of the granule before are assembled to one block of 36 samples.

Block type "**normal**"

$$z_i = x'_i \sin\left(\frac{\pi}{36}\left(i+\frac{1}{2}\right)\right) \qquad \text{for i=0 to 35}$$

Block type **"start"**

$$z_i = \begin{cases} x'_i \sin\left(\frac{\pi}{36}\left(i+\frac{1}{2}\right)\right) & \text{for i=0 to 17} \\ x'_i & \text{for i=18 to23} \\ x'_i \sin\left(\frac{\pi}{12}\left(i-18+\frac{1}{2}\right)\right) & \text{for i=24 to 29} \\ 0 & \text{for i=30 to 35} \end{cases}$$

Block type **"stop"**

$$z_i = \begin{cases} 0 & \text{for i=0 to 5} \\ x'_i \sin\left(\frac{\pi}{12}\left(i-6+\frac{1}{2}\right)\right) & \text{for i=6 to 11} \\ x'_i & \text{for i=12 to 17} \\ x'_i \sin\left(\frac{\pi}{36}\left(i+\frac{1}{2}\right)\right) & \text{for i=18 to 35} \end{cases}$$

Block type **"short"**

The block of 36 samples is divided into three overlapping blocks:

$$y_i^{(0)} = x'_{i+6} \quad \text{for i=0 to 11}$$
$$y_i^{(1)} = x'_{i+12} \quad \text{for i=0 to 11}$$
$$y_i^{(2)} = x'_{i+18} \quad \text{for i=0 to 11}$$

Each of the three small blocks is windowed separately:

$$z_i^{(k)} = y_i^{(k)} \sin\left(\frac{\pi}{12}\left(i+\frac{1}{2}\right)\right) \quad \text{for i=0 to 11, for k=0 to 2}$$

MDCT:

In the following n is the number of windowed samples. For short blocks n is 12, for long blocks n is 36. The analytical expression of the MDCT is:

$$x_i = \sum_{k=0}^{n-1} z_k \cos\left(\frac{\pi}{2n}\left(2k+1+\frac{n}{2}\right)(2i+1)\right) \qquad \text{for i=0 to } \frac{n}{2} - 1$$

Aliasing-Butterfly, Encoder:

The calculation of aliasing reduction in the encoder is performed as in the decoder. The general procedure is shown in figure A.5. The butterfly definition to be used in the encoder is shown in figure C.8. The coefficients ca_i and cs_i can be found in table B.9.

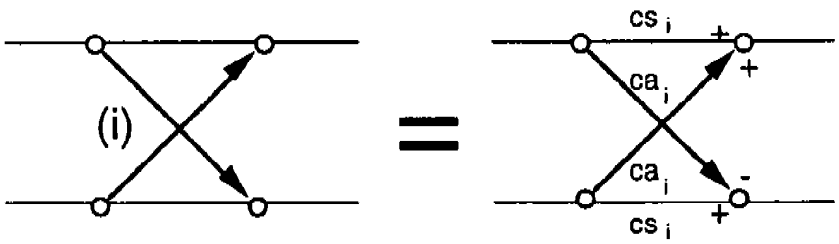

Figure C.8 -- Encoder Butterfly Definition

C.1.5.3.4 Calculation of average available bits

The average number of bits per granule is calculated from the frame size. The bitrate 64 kbits/s is used as an example. At bitrate 64 kbits/s at 48 000 samples per second,

(64 000 * (1 152/48 000) bits per frame) / (2 granules per frame) = 768 bits per granule.

As the header takes 32 bits and the side information takes 17 bytes (136 bits) in single_channel mode, the average amount of available bits for the main_data for a granule is given by

mean_bits = 768 bits per granule - (32+136 bits per frame)/(2 granules per frame) = 684 bits per granule.

Bit reservoir:

The bit reservoir can provide additional bits which may be used for the granule. The number of bits which are provided is determined within the iteration loops.

C.1.5.3.5 Quantization and encoding of frequency domain samples

The frequency domain data are quantized and coded within two nested iteration loops. Sublause C.1.5.4 contains a detailed description of these iteration loops.

C.1.5.3.6 Ancillary data

The Audio Standard provides a number of bits for the inclusion and transmission of variable length ancillary data with the audio bitstream. The ancillary data will reduce the number of bits available for audio, which may result in a degradation of audio quality.

The presence of a bit pattern in the ancillary data matching the syncword may hamper synchronization. This problem is more likely to occur when the free format is used.

C.1.5.3.7 Formatting

The details about the Layer III bitstream format can be found in 2.4.4. The formatting of the Huffman code words is described below:

The Huffman code words are in sequence from low to high frequencies. In the iteration loops the following variables have been calculated and are used in encoding the Huffman code words:

is(i), i=0...575	quantized frequency domain values
table_select[region]	Huffman code table used for regions (region = 0, 1, 2)
region_adress1	defines the border between region 0 and 1
region_adress2	defines the border between region 1 and 2
max_value[region]	maximum absolute value of quantized data in regions (region = 0, 1, 2)

The data are written to the bitstream according to the Huffman code syntax described in 2.4.2.7

The actual assembly of the Huffman code for the big_values part is described in a pseudo high level language:

```
for region number from 0 to 2
 if table_select for this region is 0
  nothing to do, all values in region are zero
 else
  if table_select for this region is > 15
   an ESC-table is used: look up linbits value connected to the table used
   for i = begin of region to end of region, count in pairs
    x = is(i), y = is(i+1)
    if x > 14
     linbitsx = x - 15, x = 15
    end if
    signx = sign(x), x = abs(x)
    if y > 14
     linbitsy = y - 15, y = 15
    end if
    signy = sign(y), y = abs(y)
   look for codeword = hcod([x][y]]) in table table_select
   write hcod([x][y]), beginning with the leftmost bit, number of bits is hlen([x][y])
    if x > 14
     write linbitsx to the bitstream, number of bits is linbits
    end if
    if x != 0
     write signx to bitstream
    end if
    if y > 14
     write linbitsy to the bitstream, number of bits is linbits
    end if
    if y != 0
     write signy to bitstream
    end if
   end do
  else
   no ESC-words are used in this region:
   for i = beginning of region to end of region, count in pairs
    x = is(i), y = is(i+1)
   signx = sign(x), x = abs(x)
   signy = sign(y), y = abs(y)
   look for codeword = hcod([x][y]) in table table_seletct
   write hcod([x][y]), beginning with the leftmost bit, number of bits is hlen([x][y])
   if x != 0
     write signx to bitstream
    end if
    if y != 0
     write signy to bitstream
    end if
   end do
  end if
 end if
end for
```

A possible application for the private_bits is to use them as frame counter.

C.1.5.4 Layer III Iteration loops

C.1.5.4.1 Introduction

The description of the Layer III loop module is subdivided into three levels. The top level is called "loops frame program". The loops frame program calls a subroutine named "outer iteration loop" which calls the subroutine "inner iteration loop". For each level a corresponding flow diagram is shown.

The loops module quantizes an input vector of spectral data in an iterative process according to several demands. The inner loop quantizes the input vector and increases the quantizer step size until the output vector can be coded with the available number of bits. After completion of the inner loop an outer loop checks the distortion of each scalefactor band and, if the allowed distortion is exceeded, amplifies the scalefactor band and calls the inner loop again.

Layer III loops module input:
(1) vector of the magnitudes of the spectral values xr(0..575).
(2) xmin(sb), the allowed distortion of the scalefactor bands. xmin = ratio(sb) * en(sb) / bw(sb).
(3) window_switching_flag which, in conjunction with mixed_block_flag and block_type, determines the number of scalefactor bands.
(4) mean_bits (bit available for the Huffman coding and the coding of the scalefactors).
(5) more_bits, the number of bits in addition to the average number of bits, as demanded by the value of the psychoacoustic entropy for the granule:
more_bits = 3.1 * PE - (average number of bits)

Layer III loops module output:
(1) vector of quantized values ix(0..575).
(2) scalefac_l(sb) or scalefac_s(sb) depending on window_switching_flag, block_type and mixed_block_flag.
(3) global_gain (quantizer step size information)
global_gain = qquant + system_constant.
system_constant includes all the scaling operations of the encoder and an offset to achieve the correct output with the decoding process described in the main part.
(4) number of unused bits available for later use.
(5) preflag (loops preemphasis on/off).
(6) Huffman code related side information
- big_values (number of pairs of Huffman coded values, excluding "count1")
- count1table_select (Huffman code table of absolut values <= 1 at the upper end of the spectrum
- table_select[0..2](Huffman code table of regions)
- region0_count, region1_count (used to calculate boundaries between regions)
- part2_3_length

C.1.5.4.2 Preparatory steps

C.1.5.4.2.1 Reset of all iteration variables

The scalefactors of the coder partitions, scalefac_l[sb] or scalefac_s[sb], are respectively set to zero.

The counter qquant for the quantizer step size is reset to zero.

Preflag is reset to zero.

Scalefac_scale is reset to zero.

The inital value of quantanf is set as follows:

quantanf = system_const * $\log_e$(sfm),

where sfm is the spectral flatness measure and quantanf depends on the computational implementation of the encoder.

The spectral flatness measure sfm is given by

$$sfm = \frac{e^{\frac{1}{n}\left(\sum_{i=0}^{n-1} \log xr(i)\right)^2}}{\frac{1}{n}\sum_{i=0}^{n-1} xr(i)^2}$$

The value of system_const is chosen so that for all signals the first iteration of the inner loop for all signals comes out with a bit sum higher than the desired bitsum. By that it is ensured that the first call of the inner loop results in the solution which uses as many of the available bits as possible. In order to spare computing time it is desirable to minimize the number of iterations by adapting the value of quantanf to the bitrate and the signal statistics.

C.1.5.4.2.2 Bit reservoir control

Bits are saved to the reservoir when fewer than the mean_bits are used to code one granule. If bits are saved for a frame, the value of main_data_end is increased accordingly. See figure A.7.a.

The number of bits which are made available for the main_data (called "max_bits") is derived from the actual estimated threshold (the PE as calculated by the psychoacoustic model), the average number of bits (mean_bits) and the actual content of the bit reservoir. The number of bytes in the bit reservoir is given by main_data_end.

The actual rules for the control of the bit reservoir are given below:

- If a number of bytes available to the inner iteration loop is not used for the Huffman encoding or other main_data, the number is added to the bit reservoir.
- If the bit reservoir contains more than 0,8 times the maximum allowed content of the bit reservoir, all bytes exceeding this number are made available for main_data (in addition to mean_bits)
- If more_bits is greater than 100 bits, then max(more_bits/8, 0,6*main_data_end) bytes are taken from the bit reservoir and made available for main_data (in addition to mean_bits).
- After the actual loops computations have been completed, the number of bytes not used for main_data is added to the bit reservoir.
- If after the step above the number of bytes in the bit reservoir exceeds the maximum allowed content, stuffing bits are written to the bitstream and the content of the bit reservoir is adjusted accordingly.

C.1.5.4.2.3 Calculation of the scalefactor selection information (scfsi)

The scfsi contains the information, which scalefactors (grouped in the scfsi_bands) of the first granule can also be used for the second granule. These scalefactors are therefore not transmitted, the bits gained can be used for the Huffman coding.

To determine the usage of the scfsi, the following information of each granule must be stored:

a) The block type

b) The total energy of the granule:

$$en\_tot = int\left\{ log2\left(\sum_{i=1}^{n} | xr(i)|^2 \right) \right\}$$

where n is the total number of spectral values.

c) The energy of each scalefactor band:

$$en(sb) = int\left\{ \log_2\left(\sum_{i=lbl(sb)}^{lbl(sb)+bw(sb)-1} | xr(i) |^2 \right) \right\}$$

where lbl(sb) is the number of the first coefficient belonging to scalefactor band sb and bw(sb) is the number of coefficients within scalefactor band sb.

d) The allowed distortion of each scalefactor band:

$$xm(sb) = \text{int}\left\{\log_2\left(xmin(i)\right)\right\}$$

xmin(sb) is calculated by the psychoacoustic model.

The scalefactors of the first granule are always transmitted. When coding the second granule, the information of the two granules is compared. There are four criteria to determine if the scfsi can be used in general. If one of the four is not fulfilled, the scfsi is disabled (that means it is set to 0 in all scfsi_bands). The criteria are (index 0 means first, index 1 second granule):

a) The spectral values are not all zero

b) None of the granules contains short blocks

c)

$$\left|en\_tot_0 - en\_tot_1\right| < en\_tot_{krit}$$

d)

$$\sum_{\text{all scalefactor bands}} \left|en(sb)_0 - en(sb)_1\right| < en\_dif_{krit}$$

If the scfsi is not disabled after the tests above, there are two criteria for each scfsi_band, which have both to be fulfilled to enable scfsi (that means to set it to 1 in this scfsi_band):

a)

$$\sum_{\text{all sc. bds in scfsi\_band}} \left|en(sb)_0 - en(sb)_1\right| < en(scfsi\_band)_{krit}$$

b)

$$\sum_{\text{all sc. bds in scfsi\_band}} \left|xm(sb)_0 - xm(sb)_1\right| < xm(scfsi\_band)_{krit}$$

The constants (with the index *krit*) have to be chosen so, that the scfsi is only enabled in case of similar energy/distortion.

Suggested values are:

en_tot_{krit}	= 10		
en_dif_{krit}	= 100		
$en(scfsi\_band)_{krit}$	=	10	for each scfsi_band
$xm(scfsi\_band)_{krit}$	=	10	for each scfsi_band

C.1.5.4.3 Outer iteration loop (distortion control loop)

The outer iteration loop controls the quantization noise which is produced by the quantization of the frequency domain lines within the inner iteration loop. The colouring of the noise is done by multiplication of the lines within scalefactor bands with the actual scalefactors before doing the quantization. The following pseudo-code illustrates the multiplication.

```
do for each scalefactor band:
    do from lower index to upper index of scale factor band
        xr(i) = xr(i) * √(2) ^ ((1 + scalefac_scale) *  scalefac(sb) )
    end do
end do
```

Where scalefac is either scalefac_l or scalefac_s as appropriate.

LUC 017280

In the actual system the multiplication is done incrementally with just the increase of the scalefactors applied in each distortion control loop. This is described in C.1.5.4.3.5 below.

The distortion loop is always started with scalefac_scale = 0. If after some iterations the maximum length of the scalefactors would be exceeded (see scalefac_compress table in 2.4.2.7 and C.1.5.4.3.5 below), then scalefac_scale is increased to the value 1 thus increasing the possible dynamic range of the scalefactors. In this case the actual scalefactors and frequency lines have to be corrected accordingly.

C.1.5.4.3.1 Saving of the scalefactors

The scalefactors of all scalefactor bands, scalefac_l(sb) or scalefac_s(sb), as well as the quantizer step size qquant are saved. If the computation of the outer loop is cancelled without having reached a proper result, this value together with the quantized spectrum give an approximation and can be transmitted.

C.1.5.4.3.2 Call of inner iteration loop

For each outer iteration loop (distortion control loop) the inner iteration loop (rate control loop) is called. The parameters are the frequency domain values (hybrid filterbank output) with the scalefactors applied to the values within the scalefactor bands and the number of bits which are available to the rate control loop. The result is the number of bits actually used and the quantized frequency lines ix(i).

C.1.5.4.3.3 Calculation of the distortion of the scalefactor bands

For each scalefactor band the actual distortion is calculated according to:

$$xfsf(sb) = \sum_{lbl(sb)}^{i=lbl(sb)+bw(sb)-1} \frac{\left(|xr(i)| - ix(i)^{4/3} * \sqrt[4]{2}^{qquant+quantanf} \right)^2}{bandwidth(sb)}$$

where lbl(sb) is the number of the coefficient representing the lowest frequency in a scalefactor band and bw(sb) is the number of coefficients within this band.

C.1.5.4.3.4 Preemphasis

The preemphasis option (switched on by setting preflag to a value of 1) provides the possibility to amplifiy the upper part of the spectrum according to the preemphasis tables, table B.6.

```
if (preflag==1) {
        ifqstep = 2 ^ (0,5*(1+scalefac_scale) )
        xmin(j) = xmin(j) *ifqstep ^ (2*prefact(j))
        for (i=lower limit of scalefactor band j; i <=upper limit of scalefactor band j; i++)  {
                xr(i) = xr(i) * ifqstep^prefact(j)
        }
}
```

The condition to switch on the preemphasis is up to the implementation. For example preemphasis could be switched on if in all of the upper 4 scalefactor bands the actual distortion exceeds the threshold after the first call of the inner loop.

If the second granule is being coded and scfsi is active in at least one scfsi_band, the preemphasis in the second granule is set equal to the setting in first granule.

C.1.5.4.3.5 Amplification of scalefactor bands which violate the masking threshold

All spectral values of the scalefactor bands which have a distortion that exceeds the allowed distortion are amplified by a factor of ifqstep. The value of ifqstep is transmitted by scalefac_scale.

```
if ( (xmin - xfsf) of scalefactor band j  <  0 ) {
        xmin(j) = xmin(j) * ifqstep ^ 2
        ifq(j) = ifq(j) + 1
        for (i=lower limit of scalefactor band; i <=upper limit of scalefactor band; i++)  {
                xr(i) = xr(i) * ifqstep
        }
}
```

If the second granule is being coded and scfsi is active in at least one scfsi_band, the following steps have to be done:

a) ifqstep has to be set similar to the first granule

b) If it is the first iteration, the scalefactors of scalefactor bands in which scfsi is enabled have to be taken over from the first granule. The corresponding spectral values have to be amplified:

```
if ( scfsi according to scalefactor band j  ==  1) {
    ifq(j) = ifq(j)first granule
    for (i=lower limit of scalefactor band; i <=upper limit of scalefactor band; i++)  {
        xr(i) = xr(i) * ifqstep ^ scalefac(j)
    }
}
```

Where scalefac is either scalefac_1 () or scalefac_s() as appropriate.

c) If it is not the first iteration, the amplification must be prevented for scalefactor bands in which scfsi is enabled.

C.1.5.4.3.6 Conditions for the termination of the loops processing

Normally the loops processing terminates if there is no scalefactor band with more than the allowed distortion. However this is not always possible to obtain. In this case there are other conditions to terminate the outer loop. If

a) All scalefactor bands are already amplified, or

b) The amplification of at least one band exceeds the upper limit which is determined by the transmission format of the scalefactors. The upper limit is a scalefactor of 15 for scalefactor bands 0 through 10 and 7 for scalefactors 11 through 20. In the case of block_type == 2 and mixed_block_flag == 0, the upper limit is 15 for scalefactors 0 through 18. In the case of block_type == 2 and mixed_block_flag == 1, the upper limit is 15 for scalefactors 0 through 17. The upper limit is 7 for other scalefactors.

The loop processing stops, and by restoring the saved sacalefac_1(sb) or scalefac_s(sb) a useful output is available. For realtime implementation, there might be a third condition added which terminates the loops in case of a lack of computing time.

C.1.5.4.4 Inner iteration loop (rate control loop)

The inner iteration loop does the actual quantization of the frequency domain data and prepares the formatting. The table selection, subdivision of the big_values range into regions and the selection of the quantizer step size takes place here.

C.1.5.4.4.1 Quantization

The quantization of the complete vector of spectral values is done according to

$$ix(i) = nint\left(\left(\frac{|xr(i)|}{\sqrt[4]{2}^{qquant+quantanf}} \right)^{0,75} - 0,0946 \right)$$

C.1.5.4.4.2 Test of the maximum of the quantized values

The maximum allowed quantized value is limited. This limit is set to constraint the table size if a table-lookup is used to requantize the quantized frequency lines. The limit is given by the possible values of the length identifier, "linbits", of values flagged with an ESC-code. Therefore before any bit counting is done the quantizer stepsize is increased by

qquant = qquant+1

until the maximum of the quantized values is within the range of the largest Huffman code table.

C.1.5.4.4.3 Calculation of the run length of zeros

The run length rzero of pairs of spectral coefficients quantized to zero on the upper end of the spectrum is counted and called "rzero".

C.1.5.4.4.4 Calculation of the run length of values less or equal one

The run length of quadrupels of spectral coefficients quantized to one or zero, following the rzero pairs of zeros, is calculated and called "count1".

C.1.5.4.4.5 Counting the bits necessary to code the values less or equal one

One Huffman code word is used to code one of the "count1" quadrupels. There are two different Huffman code books with corresponding code length tables (table A and table B in clause B.7). The number of bits to code all the count1 quadrupels is given by:

bitsum_count1 = min(bitsum_table0 , bitsum_table1)

where count1table_0 is used to point to table A

$$\text{bitsum\_table0} = \sum_{k=\text{firstcount1}}^{k=\text{firstcount1}+\text{count1}-1} \text{count1table\_0}\,(\,ix(4k)+2*ix(4k+1)+4*ix(4k+2)+8*ix(4k+3)\,)$$

and count1table_1 is used to point to table B

$$\text{bitsum\_table1} = \sum_{k=\text{firstcount1}}^{k=\text{firstcount1}+\text{count1}-1} \text{count1table\_1}\,(\,ix(4k)+2*ix(4k+1)+4*ix(4k+2)+8*ix(4k+3)\,)$$

Count1table_0 as well as count1table_1 have to include the number of bits necessary to encode the sign bits.

The information which table is used is transmitted by count1table_select, which is "0" for table A or "1" for table B, respectively.

C.1.5.4.4.6 Call of subroutine SUBDIVIDE

The number of pairs of quantized values not counted in "count1" or "rzero" is called bigvalues. SUBDIVIDE splits the scalefactor bands corresponding to this values into three groups. The last one, incomplete generally, counts as a complete one. The number of scalefactor bands in the first and second regions are contained in (region0_count+1) and (region1_count+1) respectively. The number of scalefactor bands in the third region can be calculated using bigvalues. The split strategy is up to the implementation. A very simple one for instance is to assign 1/3 of the scalefactor bands to the first and 1/4 to the last region.

Subdivide in case of blocksplit is done analoguously but there are only two subregions. Region1_count is set to a default in this case. This default is 8 in the case of split_point=0 and 9 in the case of split_point=1. Both these values point to the same absolute frequency.

C.1.5.4.4.7 Calculation of the code book for each subregion

There are 32 different Huffman code tables available for the coding of pairs of quantized values. They differ from each other in the maximum value that can be coded and in the signal statistics for which they are optimized. Only codes for values < 16 are in the table. For values >=16 there are two tables provided, where the largest value 15 is an escape character. In this case the value 15 is coded in an additional word using a linear PCM code with a word length called linbits.

A simple way to choose a table is to use the maximum of the quantized values in a subregion. tables which have the same size are optimized for different signal statistics. Therefore additional coding gain can be achieved for example by trying all of these tables.

C.1.5.4.4.8 Counting of the bits necessary to code the values in the subregions

The number of bits necessary to code the quantized values of a subregion is given by:

$$\text{bitsum}(j) = \sum_{k=0}^{k=np(j)-1} \text{bitz}\big(\text{tableselect}(j), \min(15, \text{ix}(2k+\text{fe}(j))), \min(15, \text{ix}(2k+\text{fe}(j)+1))\big)$$

$$+ \sum_{k=0}^{k=np(j)-1} \big(s(\text{ix}(2k+\text{fe}(j)) - 15) + s(\text{ix}(2k+\text{fe}(j)+1) - 15)\big) * \text{linbits}(j)$$

np(j): number of pairs in a sub region
fe(j): number of the first quantized value in a sub-region
bitz: table with Huffman code length

s(...) step function: if x >= 0 s(x) = 1
if x < 0 s(x) = 0

Note that the Huffman code length tables have to include the number of bits necessary to encode the sign bits.

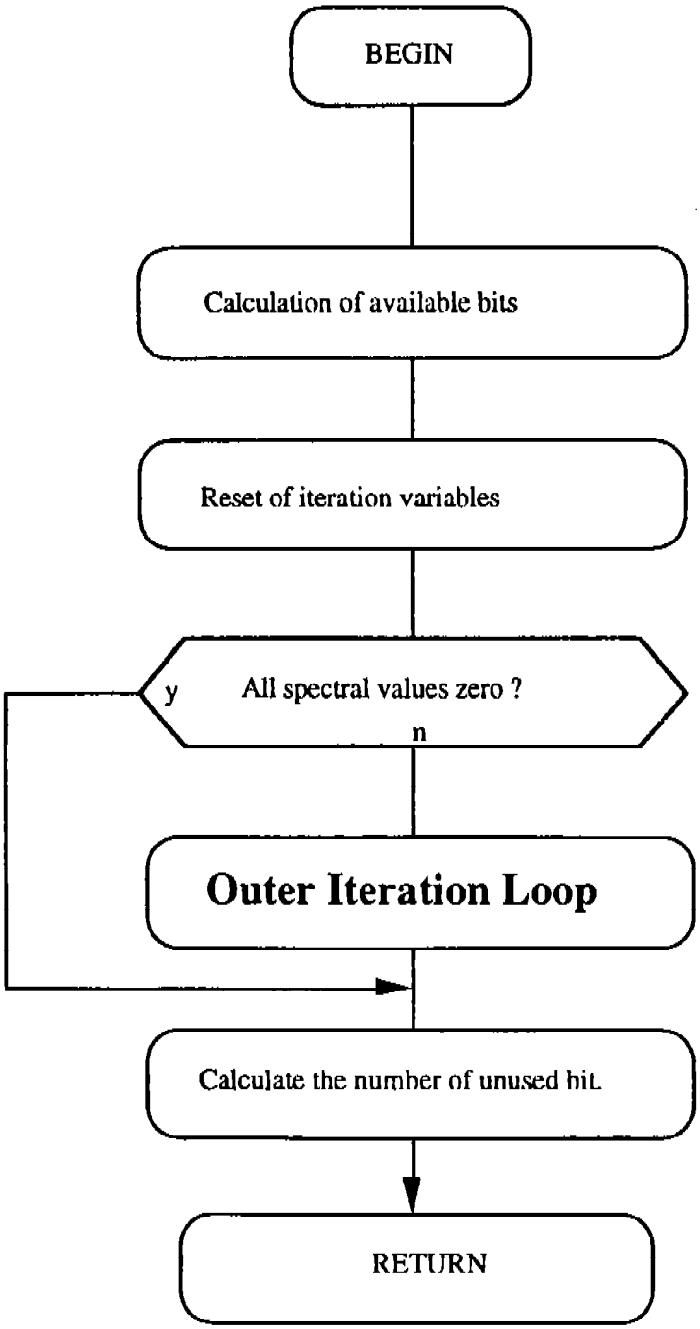

Figure C.9.a -- Layer III iteration loop

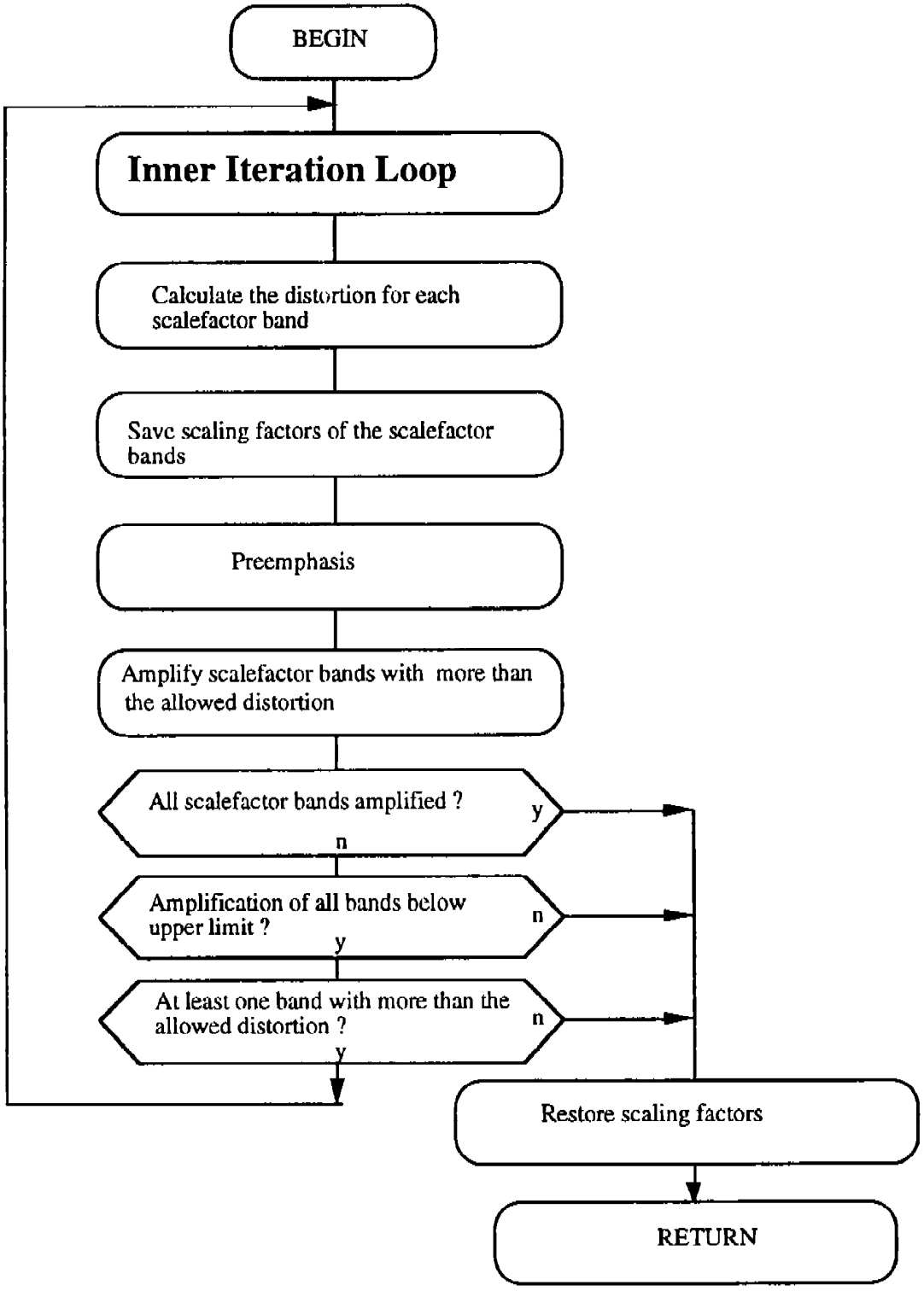

Figure C.9.b -- Layer III outer iteration loop

LUC 017286

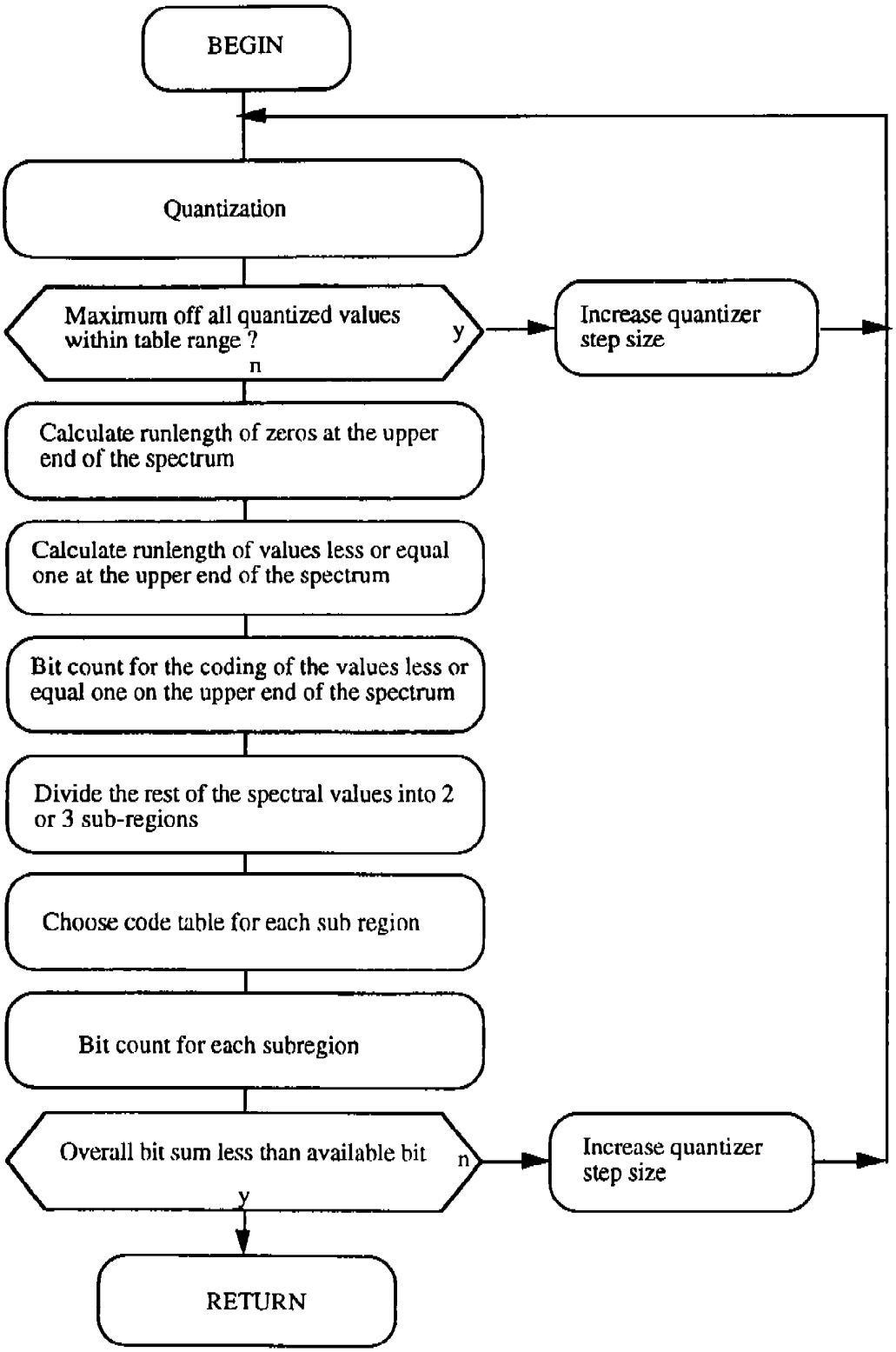

Figure C.9.c -- Layer III inner iteration loop

Annex D

(informative)

Psychoacoustic models

D.1. Psychoacoustic model 1

The calculation of the psychoacoustic model has to be adapted to the corresponding layer. This example is valid for Layers I and II. The model can be adapted to Layer III.

There is no principal difference in the application of psychoacoustic model 1 to Layer I or II.

Layer I: A new bit allocation is calculated for each block of 12 subband or 384 input PCM samples.

Layer II: A new bit allocation is calculated for three blocks totaling 36 subband samples corresponding to 3*384 (1 152) input PCM samples.

The bit allocation of the 32 subbands is calculated on the basis of the signal-to-mask ratios of all the subbands. Therefore, it is necessary to determine for each subband, the maximum signal level and the minimum masking threshold. The minimum masking threshold is derived from an FFT of the input PCM signal, followed by a psychoacoustic model calculation.

The FFT in parallel with the subband filter compensates for the lack of spectral selectivity obtained at low frequencies by the subband filterbank. This technique provides both a sufficient time resolution for the coded audio signal (Polyphase filter with optimized window for minimal pre-echoes) and a sufficient spectral resolution for the calculation of the masking thresholds. The frequencies and levels of aliasing distortions can be calculated. This is necessary for calculating a minimum bitrate for those subbands which need some bits to cancel the aliasing components in the decoder. The additional complexity to calculate the better frequency resolution is necessary only in the encoder, and introduces no additional delay or complexity in the decoder.

The calculation of the signal-to-mask-ratio is based on the following steps:

Step 1
- Calculation of the FFT for time to frequency conversion.

Step 2
- Determination of the sound pressure level in each subband.

Step 3
- Determination of the threshold in quiet (absolute threshold).

Step 4
- Finding of the tonal (more sinusoid-like) and non-tonal (more noise-like) components of the audio signal.

Step 5
- Decimation of the maskers, to obtain only the relevant maskers.

Step 6
- Calculation of the individual masking thresholds.

Step 7
- Determination of the global masking threshold.

Step 8
- Determination of the minimum masking threshold in each subband.

Step 9
- Calculation of the signal-to-mask ratio in each subband.

These steps will be further discussed. A sampling frequency of 48 kHz is assumed. For the other two sampling frequencies all frequencies mentioned should be scaled accordingly.

Step 1: FFT Analysis

The masking threshold is derived from an estimate of the power density spectrum that is calculated by a 512-point FFT for Layer I, or by a 1 024-point FFT for Layer II. The FFT is calculated directly from the input PCM signal, windowed by a Hann window.

For a coincidence in time between the bit allocation and the corresponding subband samples, the PCM-samples entering the FFT have to be delayed:

a) The delay of the analysis subband filter is 256 samples, corresponding to 5,3 ms at the 48 kHz sampling rate. A window shift of 256 samples is required to compensate for the delay in the analysis subband filter.

b) The Hann window must coincide with the subband samples of the frame. For Layer I this amounts to an additional window shift of 64 samples. For Layer II an additional window shift of minus 64 samples is required.

Technical data of the FFT:

		Layer I	Layer II
-	transform length	512 samples	1 024 samples
	Window size if fs = 48 kHz	10,67 ms	21,3 ms
	Window size if fs = 44,1 kHz	11,6 ms	23,2 ms
	Window size if fs = 32 kHz	16 ms	32 ms
-	Frequency resolution	sampling_frequency / 512	sampling_frequency / 1024

- Hann window, h(i):

$$h(i) = \sqrt{8/3} * 0{,}5 * \{1 - \cos[2 * \pi * (i)/N]\} \qquad 0 <= i <= N-1$$

- power density spectrum X(k):

$$X(k) = 10 * \log_{10} \left| \frac{1}{N} \sum_{l=0}^{N-1} h(l) * s(l) * e^{(-j*k*1*2*\pi/N)} \right|^2 \text{ dB} \qquad k = 0...N/2,$$

where s(l) is the input signal.

A normalization to the reference level of 96 dB SPL (Sound Pressure Level) has to be done in such a way that the maximum value corresponds to 96 dB.

Step 2: Determination of the sound pressure level

The sound pressure level L_{sb} in subband n is computed by:

$$L_{sb}(n) = \underset{X(k) \text{ in subband } n}{MAX}[\, X(k),\ 20*\log(scf_{max}(n)*32\,768)-10\,] \text{ dB}$$

where X(k) is the sound pressure level of the spectral line with index k of the FFT with the maximum amplitude in the frequency range corresponding to subband n. The expression $scf_{max}(n)$ is in Layer I the scalefactor, and in Layer II the maximum of the three scalefactors of subband n within a frame. The "-10 dB" term corrects for the difference between peak and RMS level. The sound pressure level $L_{sb}(n)$ is computed for every subband n.

The following alternative method of calculating $L_{sb}(n)$ offers a potential for better encoder performance, but this technique has not been subjected to a formal audio quality test.

The alternative sound pressure level L_{sb} in subband n is computed by:

$$L_{sb}(n) = MAX[\, X_{spl}(n),\ 20*\log(scf_{max}(n)*32\,768)-10\,]\ dB$$

with

$$X_{spl}(n)=10*\log_{10}\Big(\sum_{\substack{k\\ k \text{ in subband } n}} 10^{X(k)/10}\Big)\ dB$$

where $X_{spl}(n)$ is the alternative sound pressure level corresponding to subband n.

Step 3: Considering the threshold in quiet

The threshold in quiet $LT_q(k)$, also called absolute threshold, is available in the tables "Frequencies, critical band rates and absolute threshold" (tables D.1a, D.1b, D.1c for Layer I; tables D.1d, D.1e, D.1f for Layer II). These tables depend on the sampling rate of the input PCM signal. Values are available for each sample in the frequency domain where the masking threshold is calculated. An offset depending on the overall bit rate is used for the absolute threshold. This offset is -12 dB for bit rates >= 96 kbits/s and 0 dB for bit rates < 96 kbits/s per channel.

Step 4: Finding of tonal and non-tonal components

The tonality of a masking component has an influence on the masking threshold. For this reason, it is worthwhile to discriminate between tonal and non-tonal components. For calculating the global masking threshold it is necessary to derive the tonal and the non-tonal components from the FFT spectrum.

This step starts with the determination of local maxima, then extracts tonal components (sinusoids) and calculates the intensity of the non-tonal components within a bandwidth of a critical band. The boundaries of the critical bands are given in the tables "Critical band boundaries" (tables D.2a, D.2b, D.2c for Layer I; tables D.2d, D.2e, D.2f for Layer II).

The bandwidth of the critical bands varies with the center frequency with a bandwidth of about only 0,1 kHz at low frequencies and with a bandwidth of about 4 kHz at high frequencies. It is known from psychoacoustic experiments that the ear has a better frequency resolution in the lower than in the higher frequency region. To determine if a local maximum may be a tonal component, a frequency range df around the local maximum is examined. The frequency range df is given by:

Sampling rate: 32 kHz

Layer I:	df = 125 Hz	0 kHz < f <= 4,0kHz
	df = 187,5 Hz	4,0 kHz < f <= 8,0 kHz
	df = 375 Hz	8,0 kHz < f <= 15,0kHz
Layer II:	df = 62,5 Hz	0 kHz < f <= 3,0 kHz
	df = 93,75 Hz	3,0 kHz < f <= 6,0 kHz
	df = 187,5 Hz	6,0 kHz < f <= 12,0 kHz
	df = 375 Hz	12,0 kHz < f <= 24,0 kHz

Sampling rate: 44,1kHz

Layer I:	df = 172,266 Hz	0 kHz < f <= 5,512kHz
	df = 281,25 Hz	5,512 kHz < f <= 11,024 kHz
	df = 562,50 Hz	11,024 kHz < f <= 19,982kHz
Layer II:	df = 86,133 Hz	0 kHz < f <= 2,756 kHz
	df = 129,199 Hz	2,756 kHz < f <= 5,512kHz
	df = 258,398 Hz	5,512 kHz < f <= 11,024 kHz
	df = 516,797 Hz	11,024 kHz < f <= 19,982kHz

Sampling rate: 48 kHz

Layer I:	df = 187,5 Hz	0 kHz	< f <=	6,0 kHz
	df = 281,25 Hz	6,0 kHz	< f <=	12,0 kHz
	df = 562,50 Hz	12,0 kHz	< f <=	24,0 kHz
Layer II:	df = 93,750 Hz	0 kHz	< f <=	3,0 kHz
	df = 140,63 Hz	3,0 kHz	< f <=	6,0 kHz
	df = 281,25 Hz	6,0 kHz	< f <=	12,0 kHz
	df = 562,50 Hz	12,0 kHz	< f <=	24,0 kHz

To make lists of the spectral lines X(k) that are tonal or non-tonal, the following three operations are performed:

a) Labelling of local maxima

A spectral line X(k) is labelled as a local maximum if

$$X(k) > X(k-1) \quad \text{and} \quad X(k) >= X(k+1)$$

b) Listing of tonal components and calculation of the sound pressure level

A local maximum is put in the list of tonal components if

$$X(k) - X(k+j) >= 7 \text{ dB},$$

where j is chosen according to

Layer I:

j = -2, +2	for 2 < k < 63
j = -3, -2, +2, +3	for 63 <= k < 127
j = -6,..., -2, +2,..., +6	for 127<= k <= 250

Layer II:

j = -2, +2	for 2 < k < 63
j = -3, -2, +2, +3	for 63 <= k < 127
j = -6,..., -2, +2,..., +6	for 127 <= k < 255
j = -12,..., -2, +2,..., +12	for 255<= k <= 500

If X(k) is found to be a tonal component, then the following parameters are listed:

- Index number k of the spectral line.
- Sound pressure level $X_{tm}(k) = 10 * \log_{10} \{ 10^{\frac{X(k-1)}{10}} + 10^{\frac{X(k)}{10}} + 10^{\frac{X(k+1)}{10}} \}$, in dB
- Tonal flag.

Next, all spectral lines within the examined frequency range are set to $-\infty$ dB.

c) Listing of non-tonal components and calculation of the power

The non-tonal (noise) components are calculated from the remaining spectral lines. To calculate the non-tonal components from these spectral lines X(k), the critical bands z(k) are determined using the tables, "Critical band boundaries" (tables D.2a, D.2b, D.2c for Layer I; tables D.2d, D.2e, D.2f for Layer II). In Layer I, 23 critical bands are used for the sampling rate of 32 kHz, 24 critical bands for 44,1 kHz and 25 critical bands are used for 48 kHz. In Layer II, 24 critical bands are used for 32 kHz sampling rate, and 26 critical bands are used for 44,1 kHz and 48 kHz sampling rate. Within each critical band, the power of the spectral lines (remaining after the tonal components have been zeroed) are summed to form the sound pressure level of the new non-tonal component $X_{nm}(k)$ corresponding to that critical band.

The following parameters are listed:

- Index number k of the spectral line nearest to the geometric mean of the critical band.
- Sound pressure level $X_{nm}(k)$ in dB.
- Non-tonal flag.

Step 5: Decimation of tonal and non-tonal masking components

Decimation is a procedure that is used to reduce the number of maskers which are considered for the calculation of the global masking threshold.

a) Tonal $X_{tm}(k)$ or non-tonal components $X_{nm}(k)$ are considered for the calculation of the masking threshold only if:

$$X_{tm}(k) >= LT_q(k) \quad \text{or} \quad X_{nm}(k) >= LT_q(k)$$

In this expression, $LT_q(k)$ is the absolute threshold (or threshold in quiet) at the frequency of index k. These values are given in tables D.1a, D.1b, D.1c for Layer I; tables D.1d, D.1e, D.1f for Layer II.

b) Decimation of two or more tonal components within a distance of less then 0,5 Bark: Keep the component with the highest power, and remove the smaller component(s) from the list of tonal components. For this operation, a sliding window in the critical band domain is used with a width of 0,5 Bark.

In the following, the index j is used to indicate the relevant tonal or non-tonal masking components from the combined decimated list.

Step 6: Calculation of individual masking thresholds

Of the original N/2 frequency domain samples, indexed by k, only a subset of the samples, indexed by i, are considered for the global masking threshold calculation. The samples used are shown in tables D.1a, D.1b, D.1c for Layer I; tables D.1d, D.1e, D.1f for Layer II.

Layer I:
For the frequency lines corresponding to the frequency region which is covered by the first six subbands no subsampling is used. For the frequency region corresponding to the next six subbands every second spectral line is considered. Finally, in the case of 44,1 kHz and 48 kHz sampling rates, in the frequency region corresponding to the remaining subbands, every fourth spectral line is considered up to 20 kHz. In the case of 32 kHz sampling rate, in the frequency region corresponding to the remaining subbands, every fourth spectral line is considered up to 15 kHz (see also tables D.1a, D.1b, D.1c for Layer I).

Layer II:
For the frequency lines corresponding to the frequency region which is covered by the first three subbands no subsampling is used. For the frequency region which is covered by next three subbands every second spectral line is considered. For the frequency region corresponding to the next six subbands every fourth spectral line is considered. Finally, in the case of 44,1 kHz and 48 kHz sampling rates, in the remaining subbands every eighth spectral line is considered up to 20 kHz. In the case of 32 kHz sampling rate, in the frequency region corresponding to the remaining subbands, every eighth spectral line is considered up to 15 kHz. (See also tables D.1d, D.1e, D.1f for Layer II).

The number of samples, n, in the subsampled frequency domain is different depending on the sampling rates and layers.

32 kHz sampling rate:	n = 108 for Layer I	and	n = 132 for Layer II
44,1 kHz sampling rate:	n = 106 for Layer I	and	n = 130 for Layer II
48 kHz sampling rate:	n = 102 for Layer I	and	n = 126 for Layer II

Every tonal and non-tonal component is assigned the value of the index i that most closely corresponds to the frequency of the original spectral line X(k). This index i is given in tables D.1a, D.1b, D.1c for Layer I; tables D.1d, D.1e, D.1f for Layer II.

The individual masking thresholds of both tonal and non-tonal components are given by the following expression: